Memory-based Robust Interpretation of Recognised Speech

Lendvai, P.K.; van den Bosch, A.; Krahmer, E.J.; Canisius, S.V.M.

Published in:
Proceedings of the 9th International Conference "Speech and Computer", SPECOM'04

Publication date:
2004

Memory-based Robust Interpretation of Recognised Speech

Piroska Lendvai[1], *Antal van den Bosch*[1], *Emiel Krahmer*[2], *Sander Canisius*[1]

[1]ILK / Dept. Computational Linguistics and AI and [2]Dept. Communication and Cognition
Faculty of Arts, Tilburg University, The Netherlands
{P.Lendvai,Antal.vdnBosch,E.J.Krahmer,S.V.M.Canisius}@uvt.nl

Abstract

We describe a series of experiments in which memory-based machine learning techniques are used for the interpretation of spoken user input in human-machine interactions. In these experiments, the task is to determine the dialogue act of the user input and the type of information slots the user fills, on the basis of a variety of features representing the spoken input (speech measurements and word recognition information) as well as its context (the interaction history). In the first experiment, we perform this task using the complete word graph output of the automatic speech recogniser. This yields an overall accuracy of 76.2%, with an F-score of 91.3 on dialogue act classification and an F-score of 87.7 on filled slot types. In the second experiment, we investigate the usefulness of two approaches to filtering out possibly non-contributing word recognition information from the speech recogniser output: (i) filtering out disfluencies, and (ii) keeping only syntactic chunk heads.

1. Introduction

Most if not all spoken dialogue systems (SDSs) contain an understanding module that extracts semantic and pragmatic information from the *word graph* produced by the automatic speech recogniser (ASR), so that the SDS can take appropriate actions based on what the user said. These word graphs usually contain a number of hypotheses (including incorrect ones, and possibly not even including the correct one) concerning what the user just said. In many practical SDSs the understanding module employs a set of heuristic rules to extract the relevant information from these word graphs, typically by looking for specific key-words or concepts.

As an alternative, various researchers reported that combining rule-based and data-driven classification techniques can be used to interpret user input (e.g., [4, 15]), whereas the approach of [12] employs supervised learning to train on a set of utterances with their target semantic and pragmatic interpretation, and learn to associate new utterances with such interpretations. Various information sources can be used for inferring the interpretation: information from the speech recogniser, but also direct measurements derived from the acoustic signal, and the recent dialogue history.

A complicating factor is that both the user input to a SDS and the ASR output are often distorted. The ASR output often contains incorrectly recognised words, whereas the main examples of distorted (i.e., ill-formed) user input are *disfluencies* such as filled pauses, repetitions, stutters, ungrammatical constructions, and the like. In fact, the occurrence of disfluencies in user utterances can be a main stumbling block for speech recognition [7], and it is suggested that removing disfluencies might be beneficial for natural language processing [9, 18]. To filter out disfluencies from the word graph may therefore also influence the use of the speech recognition hypothesis, facilitating the automatic interpretation of user input.

Recently there is an increased interest in applying natural language processing techniques directly to the word graph output of the ASR for various language processing tasks. Some of these approaches (e.g., [21]) are generally task and language dependent, which implies that they need to be redeveloped for each new system. Others use statistical techniques, for example for sentence boundary and disfluency detection [19], recognition error correction [24], or to interpret command and control tasks [15].

In the experiments described in the current paper we perform automatic interpretation of the user input in terms of the *dialogue act(s)* expressed by the user in a turn, and the type of domain-specific pieces of information conveyed by the user, henceforth referred to as *slots* (because they represent slots of a database query, to be filled in the interaction). In a first experiment, a memory-based machine learning algorithm is supplied with the representation of the entire word graph as produced by the speech recogniser, plus a number of other features derived from the user's speech signal, as well as information from the dialogue manager.

In two additional experiments we test whether it is useful to automatically filter non-contributing or hindering material from the speech recogniser's output, aiming at a more optimal presentation of information to the classifier. The first filtering method removes potential disfluencies from the word graph automatically. The second method retains only syntactic chunk heads in the word graph, and uses those in the interpretation task. A word that is the head of its syntactic chunk generally repre-

Turn	Utterance	Annotation
S1	Good evening. This is the automatic information system of public transportation. This system provides information exclusively about train travels from a National Railways station to a National Railways station. From which station to which station do you want to travel?	Q_VA
U1	I need to go from Schiphol to Nijmegen on Tuesday next week.	S_VAD
S2	From where to where would you like to travel on Tuesday twelve December?	Q_VA;I_D
U2	From Schiphol to Nijmegen.	S_VA
S3	At what time do you want to travel from Schiphol to Nijmegen?	Q_H;I_VA
U3	Around quarter past eleven in the evening.	S_TH
S4	So you want to leave around eleven thirty-eight in the morning.	E_TH
U4	No, in the evening.	N;S_T

Figure 1: The first four annotated turns of dialogue nr. 004/005 sampled from the OVIS corpus.

sents the meaning of this chunk well: many approaches to higher-level natural language processing tasks use only syntactic heads as representation for entire chunks (cf. e.g. [2, 8]). In this experiment we test whether using syntactic heads only is sufficient for robust interpretation of spoken utterances.

The remainder of the paper is structured as follows. In section 2 we describe the data, derived from a Dutch corpus of human users communicating with a train time table information system. Section 3 describes the first experiment, where we try to classify dialogue acts and filled slot types using entire word graphs, in addition to various other features. Section 4 describes the effects of the two aforementioned techniques to filter the word graph, based on removing potential disfluencies and on removing non-head words. We conclude our study with a general discussion.

2. Data preparation

The corpus used in our study consists of 3,738 pairs of system questions and user answers; in total 441 full dialogues (involving more than 400 different speakers) sampled from the OVIS corpus [1]. Of the 3,738 user inputs, 1,613 lead to communication problems. Many of these problems are due to recognition errors, although incorrect default assumptions are also an important source of problems. The dialogues are transcribed interactions of users calling a Dutch system that provides information on train schedules in the Netherlands. The dialogues are relatively short (2-10 turns). The system uses a mixed-initiative dialogue strategy that prompts the user to fill various slots. The system needs to have these slot values before it can perform a database query. At all times, the system gives immediate feedback to the user, via implicit or explicit verification, on what it has understood. The translated dialogue in Figure 1 illustrates our material.

The system prompt and the user's response were semi-automatically labelled using a simple tag set in terms of dialogue acts and slots. Basic dialogue acts in the system prompt include asking a question (*Q*), explicit verification (*E*), repeating a prompt (*R*), asking a meta-question (*M*) and offering travel advice (final result, *Fr*). Implicit verification is represented as the simultaneous occurrence of a question and a verification (*Q;I*). The slots to be filled from the user input are departure and arrival station (*V* and *A* respectively), and the corresponding day, time of day, and hour (represented as *D*, *T* and *H* respectively).

User utterances were labelled with a similar tagset representing the dialogue acts of the utterances and the slots filled in the utterances. The dialogue acts in a user input can be the following: providing information ('slot-filling', *S*), providing an answer by explicitly uttering 'yes' (*Y*) or 'no'/'not'/'don't' (*N*), accepting incorrect information (*A*), as well as providing non-standard input (*NSTD*), e.g., by not saying anything at all. Note that a single user input can perform several dialogue acts. For instance, auser input like "Yes, to Amsterdam" is both a Yes-answer (*Y*) and a slot-filling action (*S*).

The slots the user can fill correspond to the ones the system can ask (see above). Note that whenever the user does not fill any slots in an utterance, e.g., because only an affirmative 'yes' answer is given, the slot label is marked as void. The full labelling of such input is then *Y_void*. The annotation of both system prompts and user input is illustrated in the dialogue in Figure 1. The number of different annotation labels for user utterances is 63. The number of different occurring pairs of system prompts and user responses is 480. This shows that there is no obvious, one-to-one mapping from systems prompts to user reactions. For more details on the corpus and the annotation we refer to [12].

3. Classification of dialogue acts and slots

3.1. Experimental set-up

3.1.1. Data representation

The features we employ in the classification are extracted automatically from three sources: the full word graph output of the ASR, the dialogue manager (DM) of the system, and the audio recording of the user input. The

features are listed in Table 1. From the DM we extract the words in the current and the previous system prompt, as well as the sequence of the ten most recent prompt types. The latter can be seen as a partial representation of the dialogue history, although given the overall short duration of the dialogues, the ten most recent prompts is often enough to contain the entire history. The ASR output of this particular SDS produces a word graph, containing the word hypotheses along with scores indicating for each word how confident the system is in recognising it.

Figure 2 shows the word graph for the input of the first user turn in Figure 1. It can be observed that the beginning and the ending of this turn are processed without branching in the graph, whereas in the middle several different words are hypothesised. If we unfold the word graph, it can be seen to contain eight paths ('ik moet volgende week dinsdag Schiphol naar Nijmegen' (*I need next week Tuesday Schiphol to Nijmegen*), 'ik moet volgende dinsdag Schiphol naar Nijmegen' (*I need next Tuesday Schiphol to Nijmegen*), 'ik moet volgende week dinsdag om Schiphol naar Nijmegen' (*I need next week Tuesday at Schiphol to Nijmegen*) , etc.).

We represent the recognised words (including the potentially incorrect ones) using a 759-bits, unstructured bag-of-words (BOW) vector. Each bit in that vector represents a word that occurred at least once in the entire corpus, where a '1' indicates that a certain word is present in the corresponding word graph, while a '0' indicates that it is not. For the learning task, we represent per user input, both the current and the previous word graph. Besides the BOW representation, we extract the degree of branching both from the current and the previous word graph. We determine the degree of branching by counting, for each node in the word graph, the number of outgoing edges minus one (if each node has one outgoing edge, there is no branching), and summing over these counts. The degree of branching offers a rough characterisation of the degree of confusion in the word graph; Much branching can be an indication of system uncertainty or noisy user input.

The confidence measurements in the word graph are also employed as features: for each path we sum the confidence scores over the word transitions contained in it. The path in the word graph with the highest summed confidence score is used as a basis for four features available to the learners: the highest confidence score itself, the concatenated string of words in the most confident path, the number of words in the most confident path, (as number of words may correlate with different dialogue acts, e.g., [11]), and the confidence score difference between the most confident and second-most confident path.

Various researchers have argued that prosodic information can be helpful for dialogue act classification (e.g., [10, 17]). Therefore we automatically extracted various prosodic features from the speech signal audio recordings

Aspect	Feature
DM: prompt	Sequence of last 10 prompt types
DM: lexical	Bag-of-words *(BOW)* of current prompt, BOW of previous prompt
ASR: confidence	Highest summed confidence score in current word graph, Highest summed confidence score normalised by number of nodes in path, Score difference between most confident and second-most-confident path in current word graph
ASR: branching	Branching factor in the word graph of current utterance, Branching factor in the word graph of previous utterance
ASR: lexical	BOW of current user turn, BOW of previous user turn, Word string in most confident path in current word graph, Length of most confident string
Prosody: pitch	Maximum F0, Minimum F0, Position of maximum F0, Position of minimum F0, Mean F0, Standard deviation of mean F0
Prosody: loudness	Maximum energy (RMS), Position of maximum RMS, Mean RMS, Standard deviation of mean RMS
Prosody: duration	Length of turn, Length of initial pause
Prosody: speech rate	Tempo

Table 1: Overview of the employed features.

of each user turn. In particular, we extracted (1) loudness in terms of root mean square (RMS) energy, (2) duration of the entire utterance from starting silence to ending silence, (3) pitch (in terms of F0, i.e., fundamental frequency), (4) duration of the initial pause, and (5) speech tempo. The duration of the initial pause is calculated on the basis of the most confidently recognised path in the word graph, and derived from measurement in the ASR module of the length of the silence that precedes the beginning of the speech signal. The speech tempo of the turn is measured as the number of uttered syllables per second.

As a result, each user turn is represented as a feature vector consisting of 2,482 items. Each vector also contains the combined representation of the dialogue acts and the filled slot type labels; this is the class to be predicted in the learning experiments.

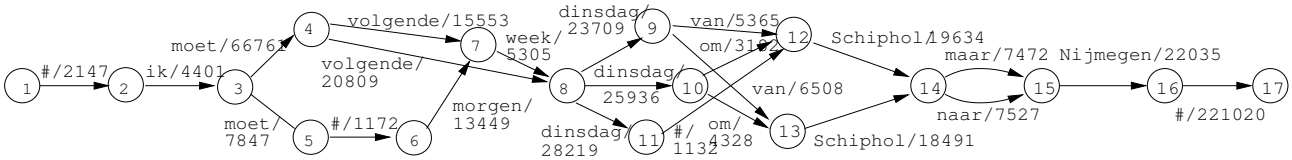

Figure 2: Word graph of the user input in turn U1 of Figure 1 *'ik moet volgende week dinsdag van Schiphol naar Nijmegen'* (I need to go from Schiphol to Nijmegen on Tuesday next week). Hash marks stand for pauses, the confidence score of each word hypothesis is given after the slash.

3.1.2. Memory-based learning experiments

We employ the IB1 algorithm, implemented in the TiMBL software package (version 5.0) [6]. The IB1 algorithm is an example of a memory-based learner (henceforth: MBL) which takes the k-nearest neighbour approach to classification by finding the most similar examples in the training data and extrapolating their class to the test instance's class. In general, there are many parameters that may influence the performance of the learning algorithm (here, for instance, the value of k, the distance weighting metric, the feature weights), and which parameter setting will yield optimal results depends on the data at hand and cannot be determined in advance. Therefore we combine IB1 with wrapped progressive sampling, a heuristic meta-learning search method that optimises algorithmic parameters automatically [20].

Training and testing of MBL is done by 10-fold cross-validation. The performance of the learner is evaluated according to four measures. Accuracy measures the percentage of correctly classified test instances. Note that this is an exact match criterion: both the dialogue act(s) and the slot(s) have to be predicted correctly. Precision, recall, and F-score are used to measure classification performance on one particular aspect of the task, i.e., on detecting the dialogue act, or the filled slot types. The F-score metric represents the harmonic mean of precision and recall [22].

The straightforward baseline of our interpretation task is to always predict the majority class label: S_VA is the most frequent label, covering 844 examples. This leads to a 22.5% overall accuracy, and already quite high scores measured in terms of precision, recall, and F-score on the individual aspects of the task: 60.4 F-score on dialogue acts, and 40.4 on identifying what slots are being filled.

3.2. Results

Table 2 contains the results of classifying dialogue acts and filled slot types in the user turn by MBL. The overall accuracy is 76.2%, which is a statistically significant improvement over the baseline ($t = 45.4, p < .01$, here and elsewhere paired t-tests are used to test for significance) and gives an error reduction of 69%. If we look at the two components of the interpretation task, we see that dialogue acts are classified with a 91.3 F-score and filled slot types with 87.7 F-score (again, both significantly higher than the baseline, with $t = 43.0, p < .01$ and $t = 37.5, p < .01$ for dialogue acts and slots respectively).

In this experiment we used the a representation of *all* the words occurring in the word graph, including the misrecognised ones. It would be interesting to see what the performance would be if we assume perfect recognition. Therefore, we ran an additional experiment in which the noisy word graph was discarded and the transcribed user utterance was used instead. We represented this transcribed utterance as a BOW, and, along with the other features, used it for the classification of dialogue acts and filled slot types. The results for this experiment give a kind of *topline*, and are provided in Table 2 as well. It can be seen that using the actual transcribed user utterance, MBL obtains an overall accuracy of 79.6, which is only 3.4 absolute points higher than the results obtained with the recognised words. The F-score for detecting dialogue acts is 92.0 points and for filled slot types the F-score is 90.5.

3.3. Discussion

The first experiment shows that a memory-based machine learning algorithm, using a representation of the entire word graph, combined with prosodic and dialogue features, can predict dialogue acts and filled slot types with relatively good accuracy. Interestingly, replacing the noisy word graph for the clean transcribed user input does not lead to a big improvement. Even though using the real, transcribed user utterance yields significantly better accuracy and F-scores, the increase is surprisingly small in view of the relatively large number of user utterances that somehow lead to communication problems. Still, there is some room for improvement, and in the next section we will see whether filtering the word graph can help. Our goal therefore is to find robust and preferably straightforward ways to present the speech recognition output in a more optimal way to the machine learning algorithm. We hypothesise that if we could automatically filter the word graph using some linguistic knowledge, it would be possible to remove potentially useless or counterproductive material from it, which might in turn lead to better performance on the interpretation task. In the remainder of

algorithm	accuracy (%)	dialogue act pre	rec	F	filled slot types pre	rec	F
Baseline	22.5	63.7	57.4	60.4	30.6	59.7	40.4
	2.2	2.0	1.8	1.8	2.2	2.7	2.4
MBL	76.2	93.5	89.3	91.3	90.7	84.9	87.7
	2.0	1.2	0.8	0.6	1.8	2.6	2.0
Topline	79.6	94.3	89.8	92.0	93.5	87.7	90.5
	1.8	1.5	1.1	1.0	1.3	1.5	1.1

Table 2: MBL's classification performance scores (with standard deviations) on the dialogue act and slot components, using the unfiltered word graph. Baseline and topline scores are given for comparison.

this paper we report on two techniques by which we aim at filtering the lexical items in the word graph.

4. Filtering the word graph

4.1. Disfluency filtering

Automatic processing of disfluent elements in speech (as well as processing of ill-formed written input) has been studied to some extent already (cf. for example [16, 18, 5] and the references cited therein). Since disfluencies in speech are known to be one source of recognition errors, it might be worthwhile to filter such elements from the word graph. To be able to do this, we first need an automatic module to *detect* disfluencies. For this purpose, we use the method of [13], who use MBL for the detection of a broad range of disfluent phenomena. A complication is that disfluencies are not systematically indicated in the transcriptions of user input in the OVIS corpus. However, after listening to the recorded speech material of the OVIS corpus we concluded that OVIS contains artificially clean transcriptions, since some user turns contain disfluent or ungrammatical items that are not transcribed in the corpus.

Our approach to disfluency filtering is therefore as follows: we first develop a disfluency detector on the basis of a large corpus of transcribed Dutch utterances, which we then apply to the word graphs from the OVIS corpus. Words in the word graph that are labelled as disfluent are filtered out, and we use this filtered result (rather than the entire word graph) for classifying dialogue acts and filled slot types.

4.1.1. Experimental set-up

Training a disfluency filter on SDC For developing the disfluency filter, we used the Spoken Dutch Corpus (SDC, [14]) of transcribed spontaneous monologues and (multi-party) dialogues. In the SDC, speech is transcribed as precisely as possible, including slips of the tongue, false starts, laughter, hesitation, background noise, etc. Figure 3 contains an example sentence from the SDC, together with its complete morpho-syntactic analysis. Note that certain leaves are not annotated as connected to the syntactic tree. These left-out leaves include a false start ('ik uh'), a filled pause ('uh'), following the word 'scepsis' (i.e., scepticism), and a repetition ('zo'n zo'n', i.e., 'such a such a'). Every word not annotated as belonging to the syntactic tree of a sentence is regarded as a disfluency.

We employed MBL to determine on the basis of the transcribed strings whether a word in that string is potentially disfluent or not, based on the left and right context (4 words to each side of the target word) as well as a variety of letter overlap features (letter overlap between consecutive words can be a good cue for various kinds of disfluencies including false starts and repetitions). The training data consists of 1,322 full discourses from SDC: 1,009,968 words in 129,932 utterances. In a 10 fold cross-validation experiment on SDC data, the method achieved an accuracy of 97% and an F-score of 80. For details, see [13].

Applying the disfluency filter on OVIS After training on the SDC, we applied the disfluency filter to all paths from all word graphs in the OVIS corpus. Looking at the hypothesised words in these paths that were classified as disfluent, we observe that the differences between the training and test data had some unfavourable effects on classification. In particular, in the SDC short sentences that consist of one or two items are most often regarded as fully disfluent (i.e., as abandoned sentences). Therefore, short user utterances, which are typical in interactions with a SDS, are often classified disfluent in the OVIS material. This is obviously wrong in many cases when for example the user answered 'yes' or 'no' to a yes/no system prompt, or provided only a station name, a day, etc. However, many repetitions and filled pauses are detected correctly in this experiment.

In total 3,818 word hypotheses are classified as disflu-

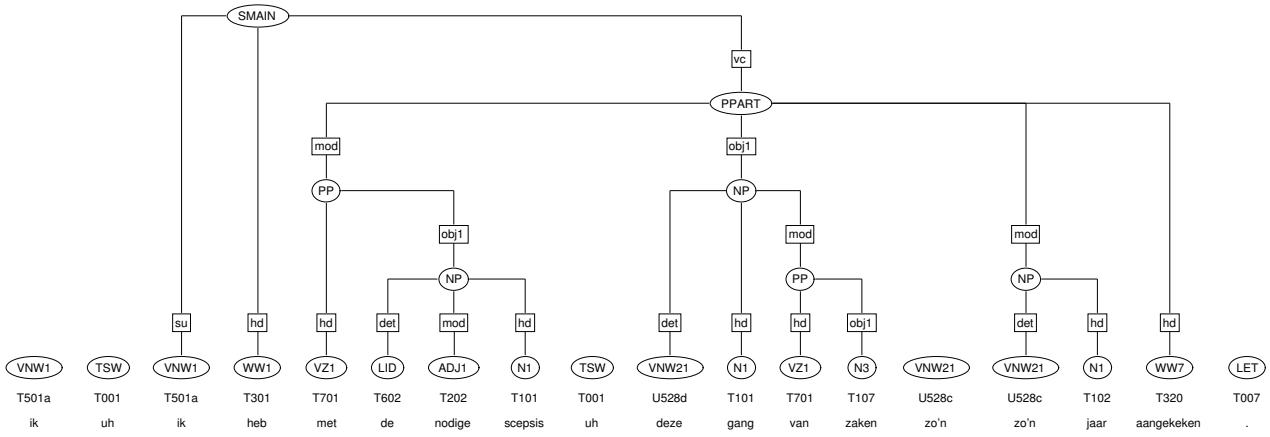

Figure 3: Example sentence with full morpho-syntactic tree from the SDC corpus: 'ik uh ik heb met de nodige scepsis uh deze gang van zaken zo'n zo'n jaar aangekeken' (I uh I have followed this process with a certain amount of scepticism uh for about a year).

ent, in each case these are removed from the BOW representation (i.e., the '1' indicating presence is switched to '0'). In all, 172 words are always classified as disfluencies, and thus are always removed from the BOW representation (i.e., are always '0'), this includes filled pauses such as 'uh' and 'uhm' (which are present in the word graph) but also various low frequency words such as rare slot values (e.g., 'jirrenveen' and 'zwaagwesteinde') or filler words (e.g., 'jazeker'; *sure*, or 'welnee'; *of course not*). Note that since these 172 words are always removed, the net effect is that the BOW vector is reduced from 759 to 587 bits. On the basis of the filtered BOW, and in combination with the other features (cf. Table 1), MBL reclassifies the user input in terms of dialogue acts and filled slots.

4.1.2. Results

The results of this learning experiment are shown in Table 3. Interpretation by MBL yields 76.3% accuracy when using the filtered word graph instead of the full word graph in learning. Compared with the scores in Table 2, the increase in accuracy of 0.1 is not statistically significant in a paired t-test. In terms of F-score, the results of dialogue act classification are somewhat lower and those of slot filling somewhat higher than the results with the unfiltered word graph (again both not statistically significantly).

4.1.3. Discussion

The results show that filtering disfluencies does not have a significant effect on the interpretation task, perhaps because the words classified disfluent receive a low feature-relevance weight in the non-filtered experiment anyway (on feature weighting in MBL cf. [6]), so that their presence is not harmful for the interpretation task. Certainly, the differences between training and testing material do not have a positive influence on this experiment.

4.2. Chunk non-head filtering

Our second method for filtering the word graph is to discard everything from the word graph except the words that act as syntactic chunk heads. For this end we use a memory-based shallow parser [3] and automatically analyse each path in the word graph.

4.2.1. Experimental set-up

Training a shallow parser on SDC As with the disfluency filter, the OVIS corpus does not provide the annotations necessary to train the memory-based shallow parser, since it is not annotated syntactically; therefore the SDC material is used for training. The SDC, a speech corpus, ought to be an appropriate source of training data for parsing the spoken utterances in the OVIS corpus. However, parsing the ASR output means parsing material that is ill-formed in possibly different ways than spontaneous speech is (cf. e.g., [24]), and this discrepancy might again lead to unfavourable inferences.

Applying the shallow parser as a non-head filter on OVIS Using the shallow parser, the paths in the word graphs are analysed syntactically: all words are assigned a complex label that encodes three types of information about the word: its part of speech tag, its syntactic chunk, and whether the word is the head word of this chunk. Below is an illustration of the parsing output of the correctly recognised path of the first user turn of our example dialogue. The chunks are indicated by square brackets; head words are marked as HD, for the rest of the labels see [23]. Note that chunks are generally quite small: many of them span

algorithm	accuracy (%)	dialogue act pre	rec	F	filled slot types pre	rec	F
DISFLUENCY FILT	76.3	93.3	88.4	90.8	92.1	85.4	88.6
	2.9	1.6	1.3	1.1	2.2	2.6	2.1
CHUNK NON-HEAD FILT	73.9	91.6	86.8	89.2	91.1	84.7	87.8
	2.7	2.5	1.3	1.7	1.2	2.8	1.7

Table 3: Classification performance of MBL on the dialogue act and slot components, using the filtered word graphs.

only a single word, which is then automatically the chunk head.

> [NP-SU-1 ik/VNW1-HD] [SMAIN-1 moet/WW1-HD] [NP volgende/WW11 week/N1-HD] [NP dinsdag/N5-HD] PNP [PP van/VZ1-HD] [NP Schiphol/N5-HD] PNP [PP naar/VZ1-HD] [NP Nijmegen/N5-HD]

For the robust interpretation task we use only the identified head words from the parser's output. Each word classified as the head of a syntactic chunk is retained, whereas all non-heads are filtered out in the same manner as was done for the disfluencies (i.e., switching the word bits from '1' to '0'). In total 577 words are classified as chunk heads in the OVIS material, effectively reducing the bag-of-words from 759 to 577 bits. On the basis of the remaining 577 words, we again create new BOW representations for each user turn. These are used together with the other word graph, DM-, and prosodic features to classify user input in terms of dialogue acts and filled slot types.

4.2.2. Results

The results of this learning experiment are shown in Table 3 as well. It can be seen that filtering non-head words leads to somewhat lower scores than not filtering at all, resulting in an accuracy of 73.8%. This is significantly lower than the result obtained by MBL without any filtering ($t = 3.1, p < .05$). On the seperate tasks, an F-score for dialogue acts of 88.6 is obtained (significantly lower than the unfiltered MBL result, $t = 4.0, p < .01$) and an F-score for filled slot types of 87.7 (identical to the unfiltered MBL result).

4.2.3. Discussion

The results show that filtering out words that are not labelled as chunk heads leads to somewhat lower accuracy and F-scores compared to not filtering. We can think of various reasons why this kind of filtering does not appear to help. One obvious limitation is the fact that the shallow parser was trained on SDC data, and applied on OVIS data, and this mismatch between training and test data is likely to hamper the results. Another potential factor might be that this particular parser produces generally small chunks, so that relatively few words are actually filtered out.

5. Concluding remarks

We described experimental results on learning to classify user input to a SDS: dialogue acts and filled slot types are detected on the basis of automatically extractable features with memory-based learning techniques. Our method attains a 91.3 F-score on identifying the dialogue act, and a 87.7 F-score on classifying filled slot types, which for both aspects is a large and significant improvement over the majority class baseline.

We then focused on the use of two filtering techniques that aim at improving this result by discarding potentially harmful words from the features of the ASR output. We tested disfluency filtering and chunk non-head filtering, carried out by memory-based learning techniques. Disfluency filtering fared somewhat better than non-head filtering, but in neither case were we able to show that filtering had a positive effect on the classification performance. We discussed a number of potential reasons for this: in both cases we were forced to use a different kind of training material from the actual test material, which has unwanted side effects in both cases. The kind of non-head filtering might be sub-optimal, since the shallow parser that we used predicts many small chunks, which implies many heads. We intend to address these issues in future work, where we also want to look at the effect of combined non-head and disfluency filtering. Finally, we also intend to train the classifiers separately on the dialogue act and slot type subtasks, which might lead to somewhat better results as well.

Given that the room for improvement available for filtering is rather small, one of the surprising findings of this study is that the results obtained using the topline filtering technique, i.e., using the actual words spoken by the user, resulted in a relatively minor improvement over those obtained using the non-filtered word graphs.

6. References

[1] L. Boves, J. Landsbergen, R. Scha, and G. van Noord. Language and Speech Technology. Research plan 1995-1997. Technical report, Nijmegen University, 1995.

[2] S. Buchholz. *Memory-based grammatical relation finding*. PhD thesis, Tilburg University, 2002.

[3] S. Canisius. Memory-based Shallow Parsing of Spoken Dutch. Master's thesis, Maastricht University, 2004.

[4] M. Cettolo, A. Corazza, and R. De Mori. A Mixed Approach to Speech Understanding. In *Proceedings of ICSLP 96*, 1996.

[5] E. Charniak and M. Johnson. Edit Detection and Parsing for Transcribed Speech. In *Proceedings of NAACL*, pages 118–126, 2001.

[6] W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. TiMBL: Tilburg Memory Based Learner, version 5.0, Reference Guide. ILK Technical Report 03-13, Tilburg University, 2003. Available from http://ilk.uvt.nl.

[7] J. Duchateau, T. Laureys, K. Demuynck, and P. Wambacq. Handling Disfluencies in Spontaneous Language Models. In *Computational Linguistics in the Netherlands 2002. Selected Papers from the Thirteenth CLIN Meeting*, pages 39–50. Rodopi, 2003.

[8] K. Hacioglu, S. Pradhan, W. Ward, J. Martin, and D. Jurafsky. Semantic Role Labeling by Tagging Syntactic Chunks. In *Proc. of CONLL*, 2004.

[9] P. Heeman and J. Allen. Detecting and Correcting Speech Repairs. In *Proc.of ACL*, pages 295–302, 1994.

[10] D. Jurafsky, E. Shriberg, B. Fox, and T. Curl. Lexical, Prosodic, and Syntactic Cues for Dialog Acts. In *Proc. of ACL/COLING-98 Workshop on Discourse Relations and Discourse Markers*, 1996.

[11] E. Krahmer, M. Swerts, M. Theune, and M. Weegels. The Dual of Denial: Two uses of disconfirmations in dialogue and their prosodic correlates. *Speech Communication*, 36(1):133–145, 2001.

[12] P. Lendvai, A. van den Bosch, and E. Krahmer. Machine Learning for Shallow Interpretation of User Utterances in Spoken Dialogue Systems. In *Proc. of EACL Workshop on Dialogue Systems: Interaction, adaptation and styles of management*, pages 69–78, 2003.

[13] P. Lendvai, A. van den Bosch, and E. Krahmer. Memory-based Disfluency Chunking. In *Proc. of DISS'03, Disfluency in Spontaneous Speech Workshop*, pages 63–66, 2003.

[14] N. Oostdijk. *The Design of the Spoken Dutch Corpus.* 2002. In: New Frontiers of Corpus Research. P. Peters, P. Collins and A. Smith (eds.), pages 105-112. Amsterdam: Rodopi.

[15] M. Rayner and B. Hockey. Transparent combination of rule-based and data-driven approaches in a speech understanding architecture. In *Proc. of EACL'03*, 2003.

[16] E. Shriberg. *Preliminaries to a theory of speech disfluencies*. PhD thesis, University of California at Berkeley, 1994.

[17] E. Shriberg, A. Stolcke, and D. Baron. Can Prosody Aid the Automatic Processing of Multi-Party Meetings? Evidence from Predicting Punctuation, Disfluencies, and Overlapping Speech. In *Proc. of ISCA Tutorial and Research Workshop on Prosody in Speech Recognition and Understanding*, pages 139–146, 2001.

[18] J. Spilker, A. Batliner, and E. Nöth. How to Repair Speech Repairs in an End-to-End System. In *Proc. ISCA Workshop on Disfluency in Spontaneous Speech*, pages 73–76, 2001.

[19] A. Stolcke, E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, G. Tur, and Y. Lu. Automatic Detection of Sentence Boundaries and Disfluencies Based on Recognized Words. In *Proc. Int. Conf. on Spoken Language Processing*, volume 5, pages 2247–2250, 1998.

[20] A. Van den Bosch. Wrapped Progressive Sampling Search for Optimizing Learning Algorithm Parameters. Technical report, ILK / Computational Linguistics and AI, Tilburg University, 2004.

[21] G. Van Noord, G. Bouma, R. Koeling, and M. Nederhof. Robust Grammatical Analysis for Spoken Dialogue Systems. *Journal of Natural Language Engineering*, 5(1):45–93, 1999.

[22] C. Van Rijsbergen. *Information Retrieval*. Buttersworth, London, 1979.

[23] T. van der Wouden, H. Hoekstra, M. Moortgat, B. Renmans, and I. Schuurman. Syntactic Analysis in the Spoken Dutch Corpus. In *Proc. LREC*, pages 768–773, 2002.

[24] K. Zechner and A. Waibel. Using chunk based partial parsing of spontaneous speech in unrestricted domains for reducing word error rate in speech recognition. In *ACL98*, 1998.